Juanita R. Brooks (CA SBN 75934) / brooks@fr.com
Roger A. Denning (CA SBN 228998) / denning@fr.com
Frank J. Albert (CA SBN 247741) / albert@fr.com
K. Nicole Williams (CA SBN 291900) / nwilliams@fr.com
Jared A. Smith (CA SBN 306576) / jasmith@fr.com
FISH & RICHARDSON P.C.
12860 El Camino Real, Ste. 400
San Diego, CA 92130
Telephone: (858) 678-5070 / Fax: (858) 678-5099

Susan E. Morrison (*Pro Hac Vice*) / morrison@fr.com
FISH & RICHARDSON P.C.
222 Delaware Avenue, 17th Floor
P.O. Box 1114
Wilmington, DE 19801
Telephone: (302) 652-5070 / Fax: (302) 652-0607
Additional counsel listed on signature page

Attorneys for Plaintiff,
FINJAN LLC

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

(SAN FRANCISCO DIVISION)

FINJAN LLC, Plaintiff, v. PALO ALTO NETWORKS, INC., Defendant.	Case No. 3:14-cv-04908-JD **FINJAN LLC'S OPPOSITION TO PALO ALTO NETWORKS, INC.'S MOTION TO STRIKE FINJAN'S INFRINGEMENT CONTENTIONS FOR THE '154, '408, AND '731 PATENTS AND TO DISMISS THESE PATENT CLAIMS WITH PREJUDICE** Date: November 17, 2022 Time: 10:00 AM Hon. James Donato Ctrm: 11, 19th Floor **REDACTED VERSION OF DOCUMENT SOUGHT TO BE SEALED**

TABLE OF CONTENTS

TABLE OF AUTHORITIES

Page(s)

Cases

I. INTRODUCTION

Finjan's infringement contentions more than meet the requirements of the Patent Local Rules. Finjan's contentions provide detailed infringement allegations on a limitation-by-limitation basis. For each limitation and accused product, the contentions (1) provide a narrative description identifying where the accused features can be found in PAN's products, (2) cite to and excerpt numerous PAN documents showing—in PAN's own words—where the infringing features are found, and (3) provide pinpoint citations to PAN's source code, showing exactly where the accused features can be found. Attempting to paint Finjan's contentions as "conclusory," "generic", and "high-level," PAN provides the Court a small fraction of Finjan's contentions, but a full view shows that the contentions are more than sufficient to put PAN on notice of Finjan's infringement positions.

Given this detail, PAN has no basis to file this motion, and its failure to explain what is missing from Finjan's source code citations is fatal to its motion. But PAN demonstrated early on that its strategy of seeking to strike Finjan's infringement contentions—***three*** times now—is not about the content of the infringement contentions, but instead is a strategy designed to paint Finjan in a negative light based on previous cases when Finjan was represented by different counsel. Indeed, PAN declared its intention to file its first motion to strike on March 11, 2021, three weeks before Finjan's initial contentions were even due or had been served. Dkt. No. 104 at pp. 23-25.

PAN's current motion is no different. PAN's new motion raises complaints that PAN never previously raised despite filing two previous motions to strike, and PAN has refused any attempt by Finjan to address its concerns. Moreover, PAN waited ***eight months*** after Finjan served its most recent amended contentions before filing this motion. Finjan served its first amended contentions on January 28, 2022. A month later, on March 1, 2022, PAN requested to meet and confer about Finjan's amended infringement contentions. *See* Exh. A. During that meet and confer on March 15, and in a letter that followed, Finjan pointed PAN to the portions of its infringement contentions laying out its infringement theories, and explained that PAN's alleged complaints about the clarity of those theories were really disputes regarding their merits. *See* Exh. B (March 22, 2022 Letter from Smith to Lin). After receiving Finjan's letter, PAN said nothing further regarding this issue for ***six months.*** Unsurprisingly, PAN failed to mention these prior communications in its motion. Mot. at p. 3.

Knowing that Finjan's infringement theories are clear, PAN instead attempts to obfuscate the contentions by pointing to narrow excerpts out-of-context, and by prematurely disputing the merits of Finjan's theories, while calling it an issue of "notice." This is not a proper motion to strike, but is instead mere gamesmanship.

II. LEGAL STANDARD

The core of Patent Local Rule 3-1 is notice. "[A]ll courts agree that the degree of specificity under Local Rule 3–1 must be sufficient to provide reasonable notice to the defendant why the plaintiff believes it has a reasonable chance of proving infringement." *Word to Info Inc v. Google Inc.*, No. 15-cv-03486-WHO, 2016 WL 3648605, at *4 (N.D. Cal. July 8, 2016). "The local rules do not require the disclosure of specific evidence nor do they require a plaintiff to prove its infringement case." *Uniloc 2017 LLC v. Apple, Inc.*, No. 19-cv-1929-EJD-VKD, 2020 WL 978678, at *2 (N.D. Cal. Feb. 28, 2020); *see also Perfect Surgical Techniques, Inc. v. Olympus Am., Inc.*, No. 12-cv-5967-PJH, 2014 WL 1095591, at *2 (N.D. Cal. Mar. 14, 2014) (similar). Because the purpose of the rule is to "provide all parties with adequate notice of and information with which to litigate their cases," the rule "distinguishes 'between the required identification of the precise element of any accused product alleged to practice a particular claim limitation, and every evidentiary *item of proof* showing that the accused element did in fact practice the limitation.'" *Comcast Cable Comms., LLC v. OpenTV, Inc.*, 2017 WL 2630088, at *3 (N.D. Cal. June 19, 2017) (quoting *AntiCancer, Inc. v. Pfizer, Inc.*, 769 F.3d 1323, 1330-31 (Fed. Cir. 2014)) (emphasis in original).

III. ARGUMENT

A. Finjan Properly Disclosed Its Theories for the '154 Patent

The '154 Patent is generally directed at inspecting inputs to software functions for potentially malicious behavior and protecting a client computer from running that software if the input is unsafe. It has wide applicability in modern computer security programs, and infringing functionality is found in PAN's Next Generation Firewall ("NGFW") products, WildFire products, Threat Prevention products, and URL Filtering products. *See* Exh. C ('154 Chart) at pp. 1-2. Because of the way that PAN integrates these products together, Finjan asserts that the following combinations of products infringe: NGFW by itself (*id.* at pp. 12-13, 44-45), NGFW with WildFire and Threat Prevention (*id.* at 13-17, 51-53), NGFW with WildFire and URL Filtering (*id.* at 17-21, 79-82), and NGFW with URL Filtering – Credential Phishing Prevention (*id.* at pp. 21-23, 95-97).[1]

Finjan's contentions provide infringement theories for each of these combinations on a limitation-by-limitation basis. For each combination, Finjan: (1) introduces all infringement theories ("Section 1"); (2) identifies documents supporting its theories ("Section 2"); (3) identifies source code supporting its theories ("Section 3"); and (4) identifies testing supporting its theories ("Section 4"). *Id.* at p. 11 (explaining how Finjan's infringement contentions are structured).

As an example, PAN's motion addresses claim element 1[a] of the '154 Patent. For the first accused configuration (NGFW by itself), Finjan's infringement theory is introduced in Section 1.1, on pages 11 through 13. In that section, the first paragraph addresses the operation of the content processor (NGFW alone), the second and third paragraphs address the operation of security computer (pattern recognition modules of NGFW), and the fourth paragraph addresses how content is handled (which includes content received over the network combined with SML files). Exh. C ('154 Chart) at pp. 12-13. The contentions address other accused configurations similarly, although in some instances the contentions identify different ways in how the aforementioned aspects of element 1[a] are addressed. *Id.* at pp. 13-17 (including multiple paragraphs addressing how content is handled in Section 1.2). Sections 2-4 identify specifics and evidence in support of those theories.

1. "Input" and "Content" Are Properly Disclosed

First, PAN complains that Finjan's contentions do not describe where the "input" and "content" limitations are found in PAN's products. This complaint is without merit. Using the

[1] Finjan also accuses PAN's Traps products of infringing the '154 Patent, '408 Patent, and '731 Patent. However, PAN does not address these separate contentions in its motion to strike.

example PAN cites in its motion, which concerns Finjan's allegations against NGFW by itself (Section 1.1), Finjan's contentions disclose that "content" is information received over a network that is combined with SML files at the NGFW. *See, e.g.*, Exh. C ('154 Chart) at p. 13 ("content requested by the client computer, which is received over a network combined with SML files received over a network at the NGFW"). PAN criticizes this disclosure as "generic" and "high-level," but in Section 2 for this limitation, Finjan's contentions identify specific examples of the same "content." *See, e.g.*, *id.* at p. 24 ("NGFWs (alone or in combination with a client computer) receive content over a network, including at least web content, flash, HTML, scripts, archive (e.g., RAR and 7-zip), binaries, documents, downloads, java, JavaScript, APIs, links, PDFs, JAR, MAC OS X, Linux files, Microsoft Office files, Android files, email, URLs, user credentials and other forms of content that can be received over a network."); *id.* at p. 42 ("The content received over a network can also include requests for credentials, entered credentials, and requests for content at a site that requested the credentials."). And then, in Section 3 for this limitation, Finjan's contentions identify source code responsible for processing the "content." *Id.* at p. 177 ("Within the PAN-OS source code, the modules that receive and process the network packets corresponding to files, URLs, and web content are implemented by source code in at least the following files: . . .").

Second, PAN complains that "files, web content, and URLs" are sometimes described as "content" and sometimes described as "inputs." Mot. at p. 5. PAN's motion feigns confusion, but Finjan's contentions are consistent with the claim language. As PAN admits, the received "content" includes an "input." *Id.* ("Per the claim language, the 'content' when received must 'include a call to a first function,' and that call must further include 'an input.'"). Thus, because the "input" is part of the claimed content, "files, web content, and URLs" are "input," but are also correctly described in Finjan's infringement theories as part of the claimed "content." The statements PAN cites from page 265 are consistent with this understanding, as they identify specific functions in PAN's source code (the specific source code files on page 264) that are involved in the processing of "files, web content, and URL." Because that source code is processing "input," it is also processing "content."

Finally, PAN states that Finjan does not explain how "an URL . . . includes 'a call to a first function' that includes an 'input.'" PAN is correct, because that is not Finjan's contention. A URL is an example of an "input," and thus is part of the claimed "content." But Finjan does not contend that a URL is a "call to first function." Instead, as set forth in Section 1 of Finjan's infringement contentions and evidenced in Sections 2 and 3, Finjan contends that the "call to a first function" is a function call within PAN's SML files:

> The accused content received over a network including a call to a first function (substitute function), the call including an input is comprised of content requested by the client computer, which is received over a network combined with SML files received over a network at the NGFW. The SML files enforce the NGFW security policies to substitute function calls into the content which cause the requested input to be sent to the security computer (pattern recognition modules) for inspection when the first function is invoked . . . . ***The call to a first function, the call including an input, varies in each instance depending on the nature of the requested content, and is implemented by at least the source code cited below and the SML files, as described below.*** The portion of the content processor for processing the content including a call to a first function and exemplary first functions are disclosed and described in Sections 3.2 - 3.8.

Exh. C ('154 Chart) at p. 13 (emphasis added).

Thus, PAN is sufficiently on notice of the claimed "content" and "input" limitations.

2. Several Exemplary "First Functions" and "Second Functions" Are Identified, and Their Operation Is Well-Detailed

PAN's argument starts with a misrepresentation of the record. PAN argues that, based on a review of Finjan's contentions, Judge Hamilton found that Finjan conceded its contentions were deficient. Mot. at p. 6. But Judge Hamilton's Order was based on statements in Finjan's ***opposition briefing***—not on a review of Finjan's contentions at that time, and certainly not on a review of the amended contentions PAN seeks to strike with the instant motion. Dkt. No. 146 at p. 2.

While PAN then complains that "Finjan fails to identify *where* in PAN's products there are specific components that constitute the [two functions]," it concedes just two paragraphs later that Finjan identifies "dozens of exemplary" functions in the source code. Indeed, Finjan identifies source code in PAN's products responsible for generating the "first function" and "second function" in Sections 3.8 and 3.9 of its infringement contentions. Exh. C ('154 Chart) at pp. 294-97 (listing exemplary first and second functions). Those identifications do not stand on their own but rather are specific examples of a "first function," such as [redacted], and a "second function" that tie to theories set forth elsewhere in Finjan's contentions. *Supra*, at pp. 4-5 (describing Finjan's contentions with respect to "input" and "content" in the context of PAN's SML files).

PAN's remaining arguments lack merit. PAN complains that Finjan's contentions "make[] no attempt at all to identify the 'content' and 'input' that are associated with the 'first functions.'" Mot. at p. 7. But as referenced above, Finjan clearly describes the claimed "content" and "input" limitations in the Accused Products, including with respect to the claimed "call to a first function." *See, e.g.*, *id.* at p. 13 ("The SML files enforce the NGFW security policies to substitute function calls into the content which cause the requested input to be sent to the security computer (pattern recognition modules) for inspection when the first function is invoked."); *see also, e.g.*, *id.* at p. 185 ("The first function that is implemented within the SML files receives the input, e.g., portions of the content requested by a webpage, executable files, executable code embedded within a webpage, executable code attached to an email, URLs embedded within a webpage, etc."). The contentions also explain how the SML files (*i.e.*, the source code that comprises a call to a first function) are "received over a network," contrary to PAN's further complaint. Mot. at p. 8; Exh. C ('154 Chart) at p. 185 ("Note that SML files are separate from PAN-OS and are received over a network, e.g., in the form of an update to the PAN-OS."); *see also, e.g.*, *id.* at p. 180 ("Besides receiving the network packets corresponding to files and URLs, the NGFW also receive network packets corresponding to PAN's State Machine Language ('SML') using, for example, the Dynamic Update process, a manual download process, or a file copy process."). Therefore, Finjan's infringement contentions consistently describe the SML files as comprising the call to a first function, operating on an input (content requested by a client computer), and received over a network at the NGFW.

PAN also wrongly complains that Finjan's infringement contentions do not describe the exemplary first functions and second functions as sharing an input. Mot. at p. 8. In discussing the very source code files that PAN cites in its motion, Finjan's contentions describe the first and second functions as sharing an input, after the input has been marked as safe by a security computer. *See, e.g.*, Exh. C ('154 Chart) at p. 202 ("once the input (files, web content, and URLs) is marked by the system, processed by the DFA pattern recognition, analyzed by the content inspection modules, and determined to be safe, the content processor transfers the input to the first function to the destination computer to be processed by the second function"), p. 209 (similar), p. 222 (similar), p. 223 (similar), p. 224 (similar). Finjan also identifies exemplary source code files responsible for "transferring the input to the first function to the destination computer to be processed by the second function." *See, e.g.*, *id.* at pp. 215-18.

Thus, Finjan has sufficiently put PAN on notice of its infringement theories with respect to the claimed "first function" and "second function."

3. The Claimed "Content Processor" and "Security Computer" Are Described Throughout Finjan's Contentions

In arguing that Finjan fails to describe the "content processor" and "security computer," PAN improperly focuses on a single introductory statement while ignoring the remainder of Finjan's contentions. Mot. at pp. 8-9. Indeed, the very next sentence in Finjan's contentions links that discussion to other relevant sections, and even cites specific source code analysis. For the infringement theory directed to NGFW alone, which PAN points to in its motion, Finjan's contentions identify specific aspects of the NGFW's content processor in Section 2.1 and again in Section 3.1. Exh. C ('154 Chart) at pp. 24-43, 177-84.

When read in total, Finjan's contentions as to the "content processor" are clear. Finjan's contentions explain that "[d]epending on the nature (e.g. file type or format) of the content, different portions of the PAN-OS source code implement the claimed content processor," and then identify source code files involved in those implementations. *Id.* at pp. 177-79 (identifying source code containing "the modules that receive and process the network packets corresponding to files, URLs, and web content"). Finjan also identifies source code files containing modules responsible for receiving SML files and processing content accordingly. *Id.* at pp. 180-84. These modules comprising the claimed content processor are implemented in NGFW's data plane. *Id.* at p. 179. Therefore, Finjan's contentions explicitly describe where to find the claimed "content processor" in PAN's Accused Products.

Similarly, Finjan's contentions properly identify the claimed "security computer" for each of its infringement theories. As PAN admits, Finjan identifies PAN's "pattern recognition modules" as the claimed "security computer" under Finjan's "NGFW" infringement theory. Mot. at pp. 8-9. Finjan explains that these modules include PAN's "Malicious Signature Matching and Deterministic Finite Automata (DFA) Matching" modules. Exh. C ('154 Chart) at p. 12. Finjan's contentions further specify where these modules may be implemented in PAN's Accused Products. *See, e.g.*, *id.* ("For NGFW's that include hardware Content Inspection, Malicious Signature Matching modules and DFA Matching modules are separate hardware components implemented on FGPAs or ASICs . . . . For NGFWs without hardware Content Inspection, Malicious Signature Matching modules and DFA Matching modules are implemented by software."). Finjan even identifies specific software modules responsible for implementing the DFA pattern recognition and content inspection hardware modules. *Id.* at pp. 206-09. Therefore, Finjan sufficiently identifies the security computer corresponding to its "NGFW" infringement theory. Finjan similarly specifies the security computers corresponding to its "WildFire," "URL Filtering," and "Credential Phishing" theories. *See, e.g.*, *id.* at pp. 51-53 (identifying WildFire or WildFire Inline ML as the security computer for the "WildFire" theory), p. 127 (identifying WildFire, PAN-DB cloud, and/or Inline Machine Learning as security computers for the "URL Filtering" theory), p. 21 (identifying WildFire, PAN-DB, Bright Cloud, Inline Machine Learning, and User-ID as security computers for the "Credential Phishing" theory).

B. Finjan Properly Disclosed its Theories for the '408 Patent

The '408 Patent is generally directed to scanning an incoming data stream to identify potential exploits while the data stream is being received. *See, e.g.*, '408 Patent at Abstract, 1:59-61. PAN implements this patent through its "stream-based" scanning functionality. That stream based functionality is found in PAN's NGFW and Threat Prevention products, and similar infringing functionality is found in PAN's WildFire and Traps products. *See* Exh. D ('408 Chart) at 1-2. As with the '154 Patent, Finjan asserts the following combinations of products infringe due to the way PAN integrates the products together: NGFW alone or in combination with WildFire and Threat Prevention (*id.* at 1, 4, 19), WildFire alone or in combination with NGFW and Threat Prevention (*id.* at 1, 4, 19), and Traps in combination with WildFire (*id.* at 1, 4).

1. Finjan Identifies "Parser Rules" and "Analyzer Rules"

PAN argues that Finjan does not "identify *where* PAN's products allegedly include 'parser rules' and 'analyzer rules'" or "explain *how* the 'parser rules' and 'analyzer rules' perform the recited functionalities relating to 'patterns' and 'tokens.'" Mot. at p. 9 (emphasis in original). But the same passage PAN quotes in its motion identifies SML files and DFA constructs as including the parser rules and analyzer rules. *Id.*

Even worse, PAN labels an excerpt from Finjan's contentions as "conclusory," but ignores that the excerpt follows an identification of specific source code files *and* an explanation as to how that source code works with pinpoint citations to source code. *Id.* at p. 125. The paragraph that PAN cites makes clear that the "scanner instantiated by the PAN-OS" includes the parser and analyzer rules in the form of SML files and DFA constructs:

> The scanner instantiated by the PAN-OS comprises parser rules and analyzer rules for the specific programming language. As an example, the PAN-OS utilizes SML files and Deterministic Finite Automata ("DFA") constructs that describe parser and analyzer rules for the specific programming language. To process the incoming stream of program code, the PAN-OS instantiates an SML virtual machine described by the corresponding SML file, thereby instantiating a scanner that comprises parser rules and analyzer rules for the specific programming language.

Id. The paragraph just prior to that identifies specific source code—by file and line number—responsible for instantiating the relevant scanner:

[redacted]

Id. at pp. 124-25. This source code disclosure is on top of the narrative explanation and documentary evidence that Finjan includes throughout its contentions. *E.g., id.* at p. 108 ("the NGFWs contain a scanner (e.g., content scanning engines) comprised of parser rules and analyzer rules for specific programming languages as shown by its usage of SML files and Deterministic

Finite Automata ('DFA') constructs that describe parser and analyzer rules for the specific programming language during content inspection process. *See, e.g.*, PAN_FIN00003766-67; PAN_FIN00133249 at 297; PAN_FIN00260837."); *see also id.* at pp. 115-19.

Thus, even though Judge Hamilton found that Finjan did not have to provide pinpoint source citations, Finjan has done so. PAN's complaint that Finjan's discussion is "conclusory" is contrary to the evidence.

PAN's complaint that Finjan does not explain "*how* the recited rules perform the recited functionalities relating to 'patterns' and 'tokens'" also lacks merit. In the claim, the "patterns" and "tokens" are lexical constructs that the scanner uses to identify "potential exploits." Consistent with the claim language, which requires that patterns and tokens are defined by the parser and analyzer rules, Finjan explains how these lexical constructs are generally defined by what Finjan identifies as the parser and analyzer rules (PAN's SML files and DFA constructs), and even identifies specific source code files as evidence:

> PAN's documentation and source code as shown below, demonstrate that the parser and analyzer rules (e.g., SML files and DFA constructs) define certain patterns in terms of tokens and identify certain combinations of tokens and patterns as being indicators of potential exploits within HTML, PowerShell, JavaScript, PDF, and Visual Basic content. See, e.g., id.: [redacted].

Exh. D ('408 Chart) at p. 108. Finjan also explains the operation of the cited source code. *Id.* at p. 132 ("[redacted].").

Finally, although PAN seeks to strike Finjan's contentions regarding parser rules and analyzer rules in their entirety, PAN does not address Finjan's contentions against PAN's WildFire product. *See generally* Mot. at pp. 9-10. PAN has waived its argument to dispute those contentions, but nonetheless, Finjan explains how PAN's source code showed "parser rules" and "analyzer rules" for specific programming languages in WildFire for its static and dynamic analyzers. *See, e.g.*, Exh. D ('408 Chart) at pp. 144, 153, 156-157, 160-66.

Thus, the Court should reject PAN's attempt to strike the entirety of Finjan's contentions for these elements.

2. Finjan Identifies a "Scanner" in Its Infringement Contentions

PAN agrees that Finjan identifies examples of "scanners," including at least NGFW's "content scanning engines" and WildFire's "Static Analyzer" and "Virtual Machine" are "scanners." Mot. at p. 10. PAN instead complains that Finjan fails to explain "how" the scanners satisfy other parts of the claim. PAN is wrong. As just one example, Finjan's contentions explain that PAN's NGFW content scanning engines (e.g., the claimed "scanner") instantiates scanners for scanning specific program languages, each of which specify parser rules and analyzer rules:

[redacted]

Exh. D ('408 Chart) at pp. 128-29. Finjan also identifies specific source code functions that parse different programming languages (e.g., HTML, JavaScript, and Visual Basic) and DFA rules for analyzing those specific programming language, which are part of the scanner. Moreover, the parties agreed that "scanner" means "software, hardware, or a combination of both for scanning." PAN cannot credibly argue that Finjan does not identify "software, hardware, or a combination of both for scanning" that is comprised of parser and analyzer rules within its accused products since Finjan identifies and explains how specific source code running in PAN's Accused Products serves as the scanner comprised of parser rules and analyzer rules (e.g., functions within the scanner source code specifying or calling the "parser rules" and "analyzer rules").

As another example that addresses the very specific "how" that PAN complains is missing, Finjan's contentions include narratives and documentation explaining how pattern matching is used by the NGFWs content inspection process to identify specific file types and exploits within those files. The contentions include documents describing how HTML, PDF, and Javascript are identified in an incoming stream using pattern matching as part of the content inspection process. Exh. D ('408 Chart) at 113 (diagram illustrating "[h]ow Content-ID works," and the use of a pattern matching system that identifies file type). The documents also show how the NGFWs content inspection process consists of the use of "Inspection Processors and Discrete Finite Automata to implement analyzer rules that identify combinations of tokes and patterns of tokens as exploits," including the involvement of PAN's [redacted] in the content inspection process. *Id.* at 115.

With respect to WildFire's static and dynamic analyzers, Finjan explains how those static and dynamic analyzers are comprised of parser and analyzer rules. For example, Finjan explains that WildFire has static analyzers that serve as scanners for specific programming languages (e.g., Java, Android, PDF), and that scanning relies on parser and analyzer rules for that specific programming language:

[redacted]

Exh. D ('408 Chart) at pp. 161-62. This is just one example of Finjan explaining where and how PAN's infringing WildFire product contained a scanner comprised of parser and analyzer rules for specific programming languages as required by the claims.

Thus, PAN's attempt to strike the entirety of Finjan's contentions with respect to the "scanner" should be rejected.

C. Finjan's Properly Disclosed Its Theories for the '731 Patent

The '731 Patent is generally directed to scanning incoming content and deriving security profiles from those scans. *See, e.g.*, '731 Patent at Abstract. As with the other patents, Finjan asserts that combinations of products infringe, including NGFW alone or in combination with WildFire and Threat Prevention (*id.* at pp. 1, 4, 24), WildFire alone or in combination with NGFW and Threat Prevention (*id.* at pp. 1, 4, 24), and Traps in combination with WildFire (*id.* at pp. 1,4).

1. Finjan Identifies a "File Cache" and "Security Profile Cache"

PAN first argues that Finjan did not identify either the "file cache" or "security profile cache" with specificity, claiming that Finjan's contentions "effectively identify every database, storage, and memory of PAN's products." But PAN's motion demonstrates the falsity in that statement, as it cites and quotes Finjan's source code analysis that identifies one of Finjan's theories for the "file cache" in the PAN products. Mot. at 12. As another example, Finjan's contentions similarly identify—by source code file—the "security profile cache." Exh. E ('731 Chart) at 153-60.

Finjan's contentions are not limited to just those caches, as Finjan identifies other databases by name where possible (otherwise by functionality, e.g., storing scanned files that are indexed by a file identifier) as the claimed "file cache" in its contentions. *See, e.g.,* Exh. E ('731 Chart) at pp. 104-05 (citing documents and source code analysis identifying relevant "caches"); *id.* at pp. 110-113, 115-125 (similar). PAN argues that Finjan's contentions are still nonspecific, but Finjan has identified those based on the information PAN has made available thus far—every identified "cache" is followed by a citation to PAN documentation or PAN source code supporting Finjan's theory relating to the relevant cache. *See id.*

PAN's complaints regarding the "security profile cache" similarly lack merit. Finjan's contentions explain *where* and *how* PAN's accused products contain a "security profile cache":

> PAN documentation for NGFW and WildFire explains that security profiles (e.g., scan results or analysis reports following a scan) are stored in a security profile cache (e.g., in a database, such as Local DB, Central DB, Virus Database, or in disk storage) after a scan ends. *See, e.g.*, PAN_FIN00249717 at 00249722; PAN_FIN00249717 at 721; PAN_FIN00010230 at 00010244; PAN_FIN00000623 at 84; PAN_FIN00008329 at 4-5; PAN_FIN00000636-638 ("The reports are available in the WildFire Submissions log on the Firewall").
>
> Moreover, as will be discussed in greater detail below, PAN documentation for NGFW and WildFire discloses that the stored security profiles (e.g., scan results or analysis reports following a scan) are indexed by a file identifier (e.g., a hash of the scanned filed), associated with a corresponding file stored in the file cache (e.g., a database, such as Local DB, or in disk storage/memory). *See, e.g.*, PAN_FIN00000636-638. FINJAN-PAN 366483 at 2
>
> As yet another example and as will be discussed in more detail below, Wildfire has a security profile cache (e.g., the DB used to store reports generated after scanning, such as Local DB, WF-DB, Central DB, or Virus DB), for storing security profiles (e.g., scan results or reports generated following a scan), which is associated with a file identifier (e.g., SHA-256 or MD5 hashes) for the corresponding stored filed. *See, .e.g.*, PAN_FIN00249717 at 00249722; PAN_FIN00008329 at 4-5; PAN_FIN00010142 at 00010155.

Exh. E ('731 Chart) at pp. 126-27.

PAN selectively picks statements from Finjan's contentions to argue that Finjan's theories are not clear. For example, PAN's quotes a small portion of Finjan's source code analysis for the NGFW and alleges that it does not "explain *how* the 'filecache1' or 'filecache2' data structures" in products satisfy the claims. However, this is false, as Finjan's contentions do explain how it contends the "filecache1" and "filecache2" within its infringing NGFW satisfy the claims. *See, e.g.*, Exh. E ('731 Chart) at pp. 118-21. Throughout its contentions for the "file cache" element, Finjan explains how PAN's products contain a "file cache" that stores files after being scanned for future access, including discussion of relevant supporting source code evidence. *See, e.g.*, Exh. E ('731 Chart) at p. 112 ("PAN's documentation shows that WildFire stores the sample (e.g., the scanned file) in a cache and the results of its analysis (security profile) in a cache."); *id.* at p. 108 ("Files that have been scanned by WildFire are subsequently cached and each file is indexed by a file identifier"). Similarly, Finjan's contentions explain in multiple places, including explanations of supporting source code evidence, how the scanners in PAN's products derive security profiles. *See, e.g.*, Exh. E ('731 Chart) at p. 131 ("upon the completion of WildFire's dynamic and static analyses, the results and protections are then delivered to the Security Platform for storage (e.g., within a security profile cache) to protect against future attacks. FINJAN-PAN 129006"); *id.* at pp. 133-34 ("PAN's documentation shows that WildFire stores the scanned samples and results of its analysis (security profile)"); *id.* at p. 137 ("The following screenshot says the "Virus Database" stores "scan results," which demonstrates that the security profiles derived by the scanner are stored in a security profile cache (e.g., Virus Database) with an identifier.").

PAN's complaint that Finjan never connects the "security profile cache" with the "file cache" ignores that Finjan provides numerous exemplary contentions and evidence showing that this limitation is satisfied. *See, e.g.*, Exh. E ('731 Chart) at p. 127 ("PAN documentation for NGFW and WildFire discloses that the stored security profiles (e.g., scan results or analysis reports following a scan) are indexed by a file identifier (e.g., a hash of the scanned filed), associated with a corresponding file stored in the file cache (e.g., a database, such as Local DB, or in disk storage/memory)."); *id.* at p. 128 ("a Behavioral Summary may contain a file identifier associated with a corresponding file in the file cache"); *id.* at p. 139 ("PAN training videos explain that NGFWs maintain security profiles derived by the scanner"); *id.* at p. 143 ("[redacted]").

Thus, the Court should reject PAN's attempt to strike the entirety of Finjan's contentions for these elements.

2. Finjan Identifies a "Security Policy Cache" in Its Contentions

PAN also raises new complaints regarding the "security policy cache" for the first time in its motion. *See* Dkt. No. 161 at pp. 13-14 (not previously raising any complaints regarding "security policy cache"). However, Finjan's contentions explain that "the security policy cache with the PAN-OS stores policies and rules set by the PAN firewall administrator that specify a list of restrictions (whether to transmit or block) for files that are transmitted to the corresponding subset of the intranet computers." Exh. E ('731 Chart) at p. 170. The contentions also identify the "security policy cache" by source code files (e.g., *id.* at pp. 169-71) and identify the types of "restrictions" (e.g., *id.* at pp. 162-63 (restrictions include whether to transmit or block, and other restrictions set forth in FINJAN-PAN 093233 and FINJAN-PAN 093574)), and show how these policies are transmitted to other computers. *E.g.*, *id.* at pp. 167-68 (citing YouTube video and PAN document that explains how policies are transmitted to other intranet computers). As such, PAN cannot credibly argue that it has "no notice of [Finjan's] infringement theory," and indeed, has admitted that Finjan's contentions state that the NGFWs store security policies (e.g., firewall administrator defined policies). Mot. at p. 14.

Thus, the Court should reject PAN's argument.

IV. CONCLUSION

For all of the above reasons, Finjan respectfully requests that the Court deny PAN's Motion to Strike Finjan's Infringement Contentions for the '154, '408, and '731 Patents. Finjan also requests that the Court deny PAN's motion to dismiss these claims with prejudice.

Dated: October 25, 2022

Respectfully Submitted,

/s/ Roger A. Denning
Juanita R. Brooks (CA SBN 75934)
brooks@fr.com
Roger A. Denning (CA SBN 228998)
denning@fr.com
Frank J. Albert (CA SBN 247741)
albert@fr.com
K. Nicole Williams (CA SBN 291900)
nwilliams@fr.com
Jared A. Smith (CA SBN 306576)
jasmith@fr.com
FISH & RICHARDSON P.C.
12860 El Camino Real, Ste. 400
San Diego, CA 92130
Telephone: (858) 678-5070 / Fax: (858) 678-5099

Aamir Kazi (*Pro Hac Vice*)
kazi@fr.com
Lawrence Jarvis (*Pro Hac Vice*)
jarvis@fr.com
FISH & RICHARDSON P.C.
1180 Peachtree St. NE, 21st floor
Atlanta, GA 30309
Telephone: (404) 892-5005 / Fax: (404) 892-5002

Susan E. Morrison (*Pro Hac Vice*)
morrison@fr.com
FISH & RICHARDSON P.C.
222 Delaware Ave., 17th Floor
P.O. Box 1114
Wilmington, DE 19801
Telephone: (302) 652-5070 / Fax: (302) 652-0607

Attorneys for Plaintiff FINJAN LLC